REMARKS

This Amendment is submitted in response to the Examiner's Action mailed June 13, 2005, with a shortened statutory period of three months set to expire September 13, 2005. Claims 1-22 are currently pending. With this amendment, claims 1, 4, 8, 9, 11, 14, 18, 19, 21, and 22 have been amended, and claims 6, 7, 10, 16, 17, and 20 have been canceled.

Applicants have amended claims 1, 11, and 21 to describe receiving a request to invoke a server application, examining an argument in the request, if the argument is an object, identifying the class that describes the object, determining whether an immutability flag that is inserted into the class that describes the object is set, wherein the object is immutable if the immutability flag inserted into the class that describes the object is set, if the object is immutable, passing an argument to the server application that includes a reference to the object, the argument not including a copy of the object, and if the object is mutable, passing a copy of the object to the server application.

Applicants have amended claims 8, 18, and 22 to describe receiving, from a caller, a request to invoke an object, if the object is immutable, passing an argument to the caller that includes a reference to the object and does not include a copy of the object, and if the object is mutable, passing a copy of the object to the caller.

Some examples of support for these amendments can be found in the specification on page 4, lines 2-14, and page 18, lines 18-27. Further support can be found in Microsoft Computer Dictionary, Fourth Edition, published 1999, which defines "class" as meaning "in object-oriented programming, a generalized category that describes a group of more specific items, called *objects*, that can exist within it."

The Examiner rejected claims 1-22 under 35 U.S.C. § 102(e) as being anticipated by U.S. Patent 6,701,520 issued to *Santosuosso*. This rejection, as it might be applied to the claims as amended, is respectfully traversed.

Santosuosso teaches elevating immutable objects to a root class such that those objects are stored in the root class. Immutable objects can also be indicated as such within an object table in the table entry for that object.

Regarding claims 1, 11, and 21, Applicants claim loading a class and inserting an immutability flag into the class itself. *Santosuosso* does not teach inserting a flag into a class.

Santosuosso teaches marking an entry in an object table to indicate that the object identified by the entry is immutable. Marking an entry that is associated with an object is not the same as inserting a flag into a class itself.

Applicants also claim receiving a request to invoke a server application where the argument of the request may be an object. This feature originally appeared in original claim 6. Regarding original claim 6, the Examiner stated that *Santosuosso* is silent on receiving such a request. Thus, the Examiner has admitted that *Santosuosso* does not teach all of the features of Applicants' clams. Therefore, because the Examiner admits that *Santosuosso* does not teach all of the features of Applicants' claims, *Santosuosso* cannot anticipate Applicants' claims.

The Examiner goes on to state that because *Santosuosso* teaches that objects are removed, there must have been a request. This is not true. Objects can be removed for a variety of reasons and in response to a variety of prompts. The mere fact that objects have been removed certainly does not imply that there must have been a request. Therefore, because *Santosuosso* does not teach a request of any sort, *Santosuosso* does not anticipate Applicants' claims.

Further, the request claimed by Applicants is a specific type of request. It is a request to invoke a server application. *Santosuosso* does not teach a server application. *Santosuosso* does not teach a request to invoke a server application. Therefore, because *Santosuosso* does not teach a request to invoke a server application, *Santosuosso* does not anticipate Applicants' claims.

Applicants claim examining an argument in the request. Again, this feature originally appeared in original claim 6. Regarding original claim 6, the Examiner stated that *Santosuosso* teaches examining an argument in Figure 3. Figure 3 teaches determining whether the class includes any elevated objects, and if so, moving them to the root class. Determining whether the class includes any elevate objects is not the same as examining an argument in a request.

Further, as discussed above, the Examiner admits that *Santosuosso* does not teach a request. Because no request is taught, there can be no argument that is taught by *Santosuosso* that is in the request. Therefore, because *Santosuosso* does not teach a request or an argument, *Santosuosso* does not teach examining an argument in a request. Because *Santosuosso* does not each examining an argument in the request, *Santosuosso* does not anticipate Applicants' claims.

According to the amended claims, Applicants claim identifying the class that describes the object and then determining whether an immutability flag that is inserted into the class that describes the object is set.

Regarding original claim 6, the Examiner stated that *Santosuosso* teaches determining whether the object is immutable if the argument is an object at column 5, lines 33-34 which state that the "Java compiler identifies any immutable objects". The mere statement that the compiler identifies any immutable objects is not the same as determining whether the object is immutable if the argument is an object. This section of *Santosuosso* does not teach what caused the Java compiler to identify immutable objects. In addition, as discussed above, *Santosuosso* does not teach an argument. Thus, *Santosuosso* does not teach determining whether the object is immutable if the argument is an object.

Applicants have further amended the claims to describe not just determining whether the object is immutable if the argument is an object, but identifying the class that describes the object and then determining whether an immutability flag that is inserted into the class that describes the object is set. *Santosuosso* does not teach determining whether an immutability flag is set because *Santosuosso* does not teach an immutability flag that is inserted into a class. *Santosuosso* does not teach determining whether an immutability flag is set where the flag is inserted into a class that describes the object. *Santosuosso* does teach identifying an immutable object, but does not teach a class that describes that object. Therefore, because *Santosuosso* does not teach identifying the class that describes the object and then determining whether an immutability flag that is inserted into the class that describes the object is set, *Santosuosso* does not anticipate Applicants' claims.

Applicants claim passing an argument to the server application that includes a reference to the object if the object is immutable where the argument does not include a copy of the object. Applicants also claim passing a copy of the object to the server application if the object is mutable.

As discussed above, *Santosuosso* does not teach a server application. Further, *Santosuosso* does not teach passing an argument to a server application.

Santosuosso does not teach passing a reference to the object if the object is immutable where the reference does not include a copy of the object. Regarding original claim 6, the Examiner stated that *Santosuosso* teaches passing a reference to the object rather than a clone of

the object if the object is immutable at column 5, lines 10-15. This section of *Santosuosso* teaches when no other object references a particular object, that particular object can be garbage collected. This section of *Santosuosso* does not describe passing a reference to the object if the object is immutable where the reference does not include a copy of the object. This section of *Santosuosso* also does not describe passing a copy of the object if the object is mutable. In fact, this section of *Santosuosso* does not teach passing a reference or an object of any type at all. Therefore, *Santosuosso* does not anticipate Applicants' claims.

Regarding claims 8, 18, and 22, these claims describe receiving, from a caller, a request to invoke an object. If the object is immutable, an argument is passed to the caller that includes a reference to the object and does not include a copy of the object. If the object is mutable, a copy of the object is passed to the caller.

As discussed above, *Santosuosso* does not teach receiving a request at all and certainly does not teach receiving a request from a caller.

As discussed above, *Santosuosso* does not teach passing an argument to the caller if the object is immutable. *Santosuosso* does not teach passing an argument to the caller that includes a reference to the object and does not include a copy of the object if the object is immutable. *Santosuosso* does not teach passing a copy of the object to the caller if the object is mutable. Therefore, *Santosuosso* does not anticipate these claims.

Because *Santosuosso* does teach each feature of Applicants' claims, *Santosuosso* does not anticipate Applicants' claims. The Examiner is invited to call the undersigned at the below-listed telephone number if in the opinion of the Examiner such a telephone conference would expedite or aid the prosecution and examination of this application.

DATE: 09.13.05

Respectfully submitted,

Lisa Yocis

Lisa L.B. Yociss
Reg. No. 36,975
Yee & Associates, P.C.
P.O. Box 802333
Dallas, TX 75380
(972) 385-8777
Attorney for Applicants